Package 'ElectDecomp'

July 21, 2025

Type Package

Title Decomposition of Seats-to-Votes Distortions

Version 0.0.1-8

Description Analyses districted electoral systems of any magnitude by computing district-party conversion ratios and seats-to-votes deviations, decomposing the sources of deviation. Traditional indexes are also computed.
References:
Kedar, O., Harsgor, L. and Sheinerman, R.A. (2016). <doi:10.1111/ajps.12225>.
Penades, A and Pavia, J.M. (2025) "The decomposition of seats-to-votes distortion in elections: mean, variance, malapportionment and participation".
Acknowledgements:
The authors wish to thank Consellería de Educación, Cultura, Universidades y Empleo, Generalitat Valenciana (grant CIACO/2023/031) for supporting this research.

License EPL

Suggests ggplot2, gridExtra

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author Alberto Penadés [aut] (ORCID: <https://orcid.org/0000-0003-1599-2105>), Jose M. Pavía [aut, cre] (ORCID: <https://orcid.org/0000-0002-0129-726X>)

Maintainer Jose M. Pavía <jose.m.pavia@uv.es>

Repository CRAN

Date/Publication 2025-01-09 12:10:05 UTC

Contents

`distortion` *Decomposition of Votes-to-Seats Distortion*

Description

Decomposes the seats-to-votes deviation for each party by means of the geographical distribution of electoral support. The deviation from proportionality is divided into two fractions: one attributable to the electoral system —further separating the mean effect and geographical variance effect— and another due to population distribution, which includes malapportionment and turnout differential effects. Additionally, the function aggregates individual party deviations into an election-wide index of deviation from proportionality (the Loosemore-Hanby index). This index is also decomposed into components attributed to the major causes of deviation from proportionality, along with interactions among them.

Usage

```
distortion(votes, seats, census)
```

Arguments

`votes`	A data.frame or matrix of order kxp (where k is the number of constituencies/districts and p the number of parties) with the votes gained by each party/candidacy in each constituency/district. When blank and/or null votes are included as columns in `votes`, they are treated in the same way as the other parties.
`seats`	A data.frame or matrix of order kxp (where k is the number of constituencies/districts and p the number of parties), with row and columns in the same order as `votes`, with the seats gained by each party/candidacy in each constituency/district.
`census`	A vector (matrix or data.frame) of length k (the number of constituencies/districts), with components in the same order as the rows in `votes`, with the census (overall population entitled to vote) in each constituency/district.

Value

A list with three components

- `inputs`: A list containing all the objects with the values used as arguments by the function.
- `district2party.contributions`: A matrix of order kxp with the district contributions to the total seat-to-vote deviation of each party. The sums across columns of this matrix yields the vector of total seat-to-vote deviations.
- `party.distortions`: A matrix of order 7xp with the estimates of the distortions at the party level.
- `aggregate.distortions`: A matrix of order 7x1 with the Loosemore-Handy index and its decomposition due to parties with and without representation as well as its cumulative decomposition step-by-step into mean effects, variance effects, malapportionment effects and turnout (and interaction) effects.

Author(s)

Alberto Penades, <alberto.penades@gmail.com>

Jose M. Pavia, <pavia@uv.es>

References

Penades, A and Pavia, JM. *The decomposition of seats-to-votes distortion in elections: mean, variance, malapportionment and participation*.

Examples

```
votes.ex <- structure(list(Party1 = c(92019L, 112927L, 117482L, 152498L, 89494L,
                                      103697L, 115390L, 223252L, 55837L, 46820L,
                                      138575L, 177843L, 163536L, 187254L, 140237L,
                                      102719L, 64603L, 102362L, 70550L, 39321L,
                                      101891L, 71650L, 89363L, 137495L, 51568L,
                                      108862L, 50270L, 35324L, 104145L, 60626L,
                                      355857L, 45231L, 45191L, 69014L, 148697L,
                                      115729L, 221996L, 87059L, 99440L, 198231L,
                                      737699L, 75036L, 38338L, 0L, 91262L, 181633L,
                                      56917L, 197100L, 84115L, 334705L, 8808L, 10723L),
                           Party2 = c(50723L, 150896L, 122361L, 111746L, 63172L,
                                      123708L, 185095L, 251336L, 33693L, 16423L,
                                      111293L, 182850L, 73554L, 39616L, 50951L,
                                      67611L, 56332L, 79715L, 28350L, 17133L, 83956L,
                                      14984L, 44388L, 64766L, 25878L, 44168L, 18210L,
                                      10757L, 76308L, 26225L, 721880L, 60747L, 27809L,
                                      59926L, 107797L, 54827L, 78598L, 20808L, 21079L,
                                      54642L, 731380L, 54720L, 34244L, 93010L, 140643L,
                                      155871L, 36186L, 213242L, 69976L, 395211L,
                                      7886L, 5186L),
                            Party3 = c(11926L, 41809L, 59668L, 33879L, 10217L, 29847L,
                                      50990L, 91914L, 7398L, 2442L, 22446L, 60297L,
                                      14046L, 7467L, 10643L, 13971L, 13623L, 15562L,
                                      7927L, 5521L, 21942L, 2406L, 4765L, 12460L, 3744L,
                                      5522L, 2101L, 1196L, 15798L, 2783L, 469361L,
                                      24746L, 22680L, 41345L, 21932L, 6797L, 16777L,
                                      2949L, 2627L, 11835L, 247038L, 6319L, 3906L,
                                      12042L, 29968L, 29840L, 3846L, 50444L, 14029L,
                                      106133L, 0L, 966L),
                             Party4 = c(14886L, 20424L, 33616L, 24573L, 9415L, 26422L,
                                      34838L, 42860L, 7223L, 15180L, 34857L, 77932L,
                                      28472L, 15944L, 27828L, 36598L, 16005L, 31310L,
                                      10277L, 12815L, 44091L, 7123L, 28707L, 33285L,
                                      14638L, 15259L, 7407L, 3792L, 20521L, 30677L,
                                      75097L, 7953L, 10067L, 15216L, 21880L, 19516L,
                                      50256L, 36377L, 21502L, 40104L, 242077L, 0L,
                                      7927L, 27048L, 36934L, 30167L, 19925L, 35755L,
                                      14596L, 60410L, 2915L, 2074L)),
                              class = "data.frame", row.names = c(NA, -52L))
```

```
seats.ex <- structure(list(Party1 = c(3L, 2L, 3L, 4L, 3L, 3L, 3L, 5L, 2L, 2L, 3L,
                                      4L, 4L, 5L, 5L, 3L, 2L, 3L, 3L, 2L, 2L, 3L,
                                      3L, 4L, 2L, 3L, 2L, 3L, 3L, 2L, 5L, 1L, 1L,
                                      2L, 4L, 4L, 6L, 4L, 4L, 6L, 11L, 3L, 2L, 0L,
                                      2L, 4L, 2L, 4L, 2L, 5L, 1L, 1L),
                           Party2 = c(2L, 4L, 3L, 3L, 2L, 4L, 4L, 5L, 1L,
                                      1L, 3L, 4L, 2L, 1L, 2L, 1L, 2L, 2L, 1L, 1L,
                                      2L, 0L, 1L, 1L, 1L, 1L, 1L, 0L, 2L, 1L, 11L,
                                      2L, 1L, 1L, 3L, 1L, 2L, 0L, 0L, 1L, 11L, 2L,
                                      1L, 3L, 3L, 4L, 1L, 4L, 2L, 7L, 0L, 0L),
                           Party3 = c(0L, 1L, 1L, 0L, 0L, 0L, 1L, 2L, 0L, 0L, 0L,
                                      1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
                                      0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 7L, 0L, 0L,
                                      1L, 0L, 0L, 0L, 0L, 0L, 0L, 4L, 0L, 0L, 0L,
                                      0L, 0L, 0L, 1L, 0L, 1L, 0L, 0L),
                           Party4 = c(0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
                                      1L, 0L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 1L, 0L,
                                      0L, 1L, 0L, 0L, 0L, 0L, 0L, 1L, 1L, 0L, 0L,
                                      0L, 0L, 0L, 1L, 1L, 1L, 1L, 3L, 0L, 0L, 0L,
                                      1L, 0L, 1L, 0L, 0L, 1L, 0L, 0L)),
                           class = "data.frame", row.names = c(NA, -52L))
 census.ex <- c(239935L, 559590L, 451021L, 456952L, 255557L, 403142L, 569535L,
                842042L, 153109L, 112376L, 547396L, 783206L, 407899L, 365063L,
                397732L, 327008L, 211249L, 310547L, 151099L, 95061L, 309595L,
                130998L, 238885L, 364127L, 125385L, 241301L, 99408L, 73992L,
                295410L, 164706L, 3004988L, 304448L, 242148L, 331744L, 413193L,
                279372L, 731499L, 302293L, 315033L, 548711L, 2744152L, 319222L,
                153080L, 437592L, 749572L, 551021L, 167030L, 677156L, 283161L,
                1287981L, 31672L, 25368L)

example <- distortion(votes.ex, seats.ex, census.ex)
example$party.distortions
```

`inequality` *Inequality in votes-to-seats conversion*

Description

Computes the votes-to-seats conversion ratios for each combination of district and party in order to draw the representational inequality (Lorenz) curve of an election (or multiple elections under one electoral system). The Representational Inequality index proposed by Kedar, Harsgor, and Sheinerman, (2016) is calculated as a system-level index based on the area under the curve. (Use `plot` on its output to draw the curve.)

Usage

```
inequality(votes, seats)
```

Arguments

votes	A data.frame or matrix of order kxp (where k is the number of constituencies/districts and p the number of parties) with the votes gained by each party/candidacy in each constituency/district. When blank and/or null votes are included as columns in votes, they are treated in the same way as the other parties.
seats	A data.frame or matrix of order kxp (where k is the number of constituencies/districts and p the number of parties), with row and columns in the same order as votes, with the seats gained by each party/candidacy in each constituency/district.

Value

A list with four components

- inputs: A list containing all the objects with the values used as arguments by the function.
- conversion.ratios: A data.frame of order Nx7 (where N <= k*p) with conversion ratios of votes-to-seats for each combination of district and party. The first and second columns identify the district and party, respectively. The third column contains the conversion ratios. The last two columns (which are derived from the forth and fifth columns) present the cumulative proportions of seats and votes, from which the representational inequality curve of the election can be drawn.
- RI: The estimated Representational Inequality (RI) Index, which equals two times the area between the 45-degree line of perfect equality in votes conversion and the (Lorenz) representational inequality curve.
- other.statistics: A matrix of order 3x1 with the total election proportions of non-represented, underrepresented and overrepresented votes.

Author(s)

Jose M. Pavia, <pavia@uv.es>

Alberto Penades, <alberto.penades@gmail.com>

References

Kedar, O, Harsgor, L and Sheinerman, RA (2016). Are voters equal under proportional representation? *American Journal of Political Science*, 60(3), 676-691 doi:10.1111/ajps.12225

Examples

```
votes.ex <- structure(list(Party1 = c(92019L, 112927L, 117482L, 152498L, 89494L,
                                      103697L, 115390L, 223252L, 55837L, 46820L,
                                      138575L, 177843L, 163536L, 187254L, 140237L,
                                      102719L, 64603L, 102362L, 70550L, 39321L,
                                      101891L, 71650L, 89363L, 137495L, 51568L,
                                      108862L, 50270L, 35324L, 104145L, 60626L,
                                      355857L, 45231L, 45191L, 69014L, 148697L,
                                      115729L, 221996L, 87059L, 99440L, 198231L,
                                      737699L, 75036L, 38338L, 0L, 91262L, 181633L,
```

```
                                  56917L, 197100L, 84115L, 334705L, 8808L, 10723L),
                          Party2 = c(50723L, 150896L, 122361L, 111746L, 63172L,
                                  123708L, 185095L, 251336L, 33693L, 16423L,
                                  111293L, 182850L, 73554L, 39616L, 50951L,
                                  67611L, 56332L, 79715L, 28350L, 17133L, 83956L,
                                  14984L, 44388L, 64766L, 25878L, 44168L, 18210L,
                                  10757L, 76308L, 26225L, 721880L, 60747L, 27809L,
                                  59926L, 107797L, 54827L, 78598L, 20808L, 21079L,
                                  54642L, 731380L, 54720L, 34244L, 93010L, 140643L,
                                  155871L, 36186L, 213242L, 69976L, 395211L,
                                  7886L, 5186L),
                           Party3 = c(11926L, 41809L, 59668L, 33879L, 10217L, 29847L,
                                  50990L, 91914L, 7398L, 2442L, 22446L, 60297L,
                                  14046L, 7467L, 10643L, 13971L, 13623L, 15562L,
                                  7927L, 5521L, 21942L, 2406L, 4765L, 12460L, 3744L,
                                  5522L, 2101L, 1196L, 15798L, 2783L, 469361L,
                                  24746L, 22680L, 41345L, 21932L, 6797L, 16777L,
                                  2949L, 2627L, 11835L, 247038L, 6319L, 3906L,
                                  12042L, 29968L, 29840L, 3846L, 50444L, 14029L,
                                  106133L, 0L, 966L),
                            Party4 = c(14886L, 20424L, 33616L, 24573L, 9415L, 26422L,
                                  34838L, 42860L, 7223L, 15180L, 34857L, 77932L,
                                  28472L, 15944L, 27828L, 36598L, 16005L, 31310L,
                                  10277L, 12815L, 44091L, 7123L, 28707L, 33285L,
                                  14638L, 15259L, 7407L, 3792L, 20521L, 30677L,
                                  75097L, 7953L, 10067L, 15216L, 21880L, 19516L,
                                  50256L, 36377L, 21502L, 40104L, 242077L, 0L,
                                  7927L, 27048L, 36934L, 30167L, 19925L, 35755L,
                                  14596L, 60410L, 2915L, 2074L)),
                             class = "data.frame", row.names = c(NA, -52L))

seats.ex <- structure(list(Party1 = c(3L, 2L, 3L, 4L, 3L, 3L, 3L, 5L, 2L, 2L, 3L,
                                  4L, 4L, 5L, 5L, 3L, 2L, 3L, 3L, 2L, 2L, 3L,
                                  3L, 4L, 2L, 3L, 2L, 3L, 3L, 2L, 5L, 1L, 1L,
                                  2L, 4L, 4L, 6L, 4L, 4L, 6L, 11L, 3L, 2L, 0L,
                                  2L, 4L, 2L, 4L, 2L, 5L, 1L, 1L),
                            Party2 = c(2L, 4L, 3L, 3L, 2L, 4L, 4L, 5L, 1L,
                                  1L, 3L, 4L, 2L, 1L, 2L, 1L, 2L, 2L, 1L, 1L,
                                  2L, 0L, 1L, 1L, 1L, 1L, 1L, 0L, 2L, 1L, 11L,
                                  2L, 1L, 1L, 3L, 1L, 2L, 0L, 0L, 1L, 11L, 2L,
                                  1L, 3L, 3L, 4L, 1L, 4L, 2L, 7L, 0L, 0L),
                            Party3 = c(0L, 1L, 1L, 0L, 0L, 0L, 1L, 2L, 0L, 0L, 0L,
                                  1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
                                  0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 7L, 0L, 0L,
                                  1L, 0L, 0L, 0L, 0L, 0L, 0L, 4L, 0L, 0L, 0L,
                                  0L, 0L, 0L, 1L, 0L, 1L, 0L, 0L),
                            Party4 = c(0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
                                  1L, 0L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 1L, 0L,
                                  0L, 1L, 0L, 0L, 0L, 0L, 0L, 1L, 1L, 0L, 0L,
                                  0L, 0L, 0L, 1L, 1L, 1L, 1L, 3L, 0L, 0L, 0L,
                                  1L, 0L, 1L, 0L, 0L, 1L, 0L, 0L)),
                            class = "data.frame", row.names = c(NA, -52L))
```

```
example <- inequality(votes.ex, seats.ex)
example$RI
```

`malapportionment_index` *Malapportionment index*

Description

Computes the malapportionment index proposed in Samuels and Snyder (2001).

Usage

```
malapportionment_index(seats, census)
```

Arguments

`seats`	A vector (matrix or data.frame) of length k (the number of constituencies/districts), components in the same order as `census`, with the seats apportioned in each constituency/district.
`census`	A vector (matrix or data.frame) of length k (the number of constituencies/districts), with components in the same order as `seats`, with the census (overall population entitled to vote) in each constituency/district.

Value

The Value of the malapportionment index.

Author(s)

Alberto Penades, <alberto.penades@gmail.com>

Jose M. Pavia, <pavia@uv.es>

References

Samuels, D and Snyder, R (2001) The Value of a Vote: Malapportionment in Comparative Perspective. *British Journal of Political Science*, 31 , 651-671.

Examples

```
seats.ex <- c(5, 7, 7, 7, 5, 7, 8, 12, 3, 3, 6, 10, 6, 6, 7, 5, 4, 5, 4,
              3, 5, 3, 4, 6, 3, 4, 3, 3, 5, 4, 24, 3, 2, 4, 7, 5, 9, 5, 5,
              8, 29, 5, 3, 3, 6, 8, 4, 9, 4, 14, 1, 1)
census.ex <- c(239935L, 559590L, 451021L, 456952L, 255557L, 403142L, 569535L,
               842042L, 153109L, 112376L, 547396L, 783206L, 407899L, 365063L,
               397732L, 327008L, 211249L, 310547L, 151099L, 95061L, 309595L,
               130998L, 238885L, 364127L, 125385L, 241301L, 99408L, 73992L,
```

```
                295410L, 164706L, 3004988L, 304448L, 242148L, 331744L, 413193L,
                279372L, 731499L, 302293L, 315033L, 548711L, 2744152L, 319222L,
                153080L, 437592L, 749572L, 551021L, 167030L, 677156L, 283161L,
                1287981L, 31672L, 25368L)

malapportionment_index(seats.ex, census.ex)
```

`plot.ElectDecomp` *Graphical representation of ElectDecomp objects*

Description

Plot method for objects obtained with the functions `distortion` and `inequality` of the ElectDecomp package.

Usage

```
## S3 method for class 'ElectDecomp'
plot(
  x,
  type = "decomposition",
  panels.size = "global",
  title.size = 12,
  axis.title.size = 12,
  summaries = "RI",
  bar.width = 0.2,
  bar.color = "skyblue",
  text.color = "black",
  text.size = 3,
  labels.size = 10,
  ticks.linewidth = 0.2,
  length.breaks = 0.02,
  labels.y.size = 9,
  labels.x.size,
  parties,
  party.names,
  panels.title,
  axis.title,
  ...,
  show.plot = TRUE
)
```

Arguments

`x`	An object output of either the `distortion` or `inequality` functions of the ElectDecomp package.

`type` When x is an output of `distortion`, this argument informs the output to be plotted. If `type = "districts"`, the method plots the `district2party.contributions` matrix; otherwise it plots the decomposition of the total deviation available in the `party.distortions` matrix.

`panels.size` When plotting an output of `distortion`, this argument informs about how the widths of panels should be calculated: either considering the deviations of all the parties ("global") or only the deviations of the plotted parties ("local"). Default, `"global"`.

`title.size` Size of the text of the panel titles. Default, 12.

`axis.title.size` Size of the text of the axis titles. Default, 12.

`summaries` This argument Informs whether some summary statistics about the election should be included in the plot: "none", no summary is included; "all" RI and `other.statistics` are included; "RI", only the representational inequality (RI) index is included. Default, "RI". Only active when an output of the `inequality` function is to be plotted.

`bar.width` Width of the bars or of the inequality curve. Default, 0.2.

`bar.color` Color of the bars or of the inequality curve. Default, "skyblue".

`text.color` Color to be used for text inside the plot. Default, "black".

`text.size` Size to be used for text inside the plot. Default, 3.

`labels.size` Size to be used for the labels of the axis. Default, 10.

`ticks.linewidth` Width of the lines to be used as ticks. Default, 0.2. Only active when the `party.distortions` matrix output of the `distortion` function is to be plotted.

`length.breaks` Distance between labels of the breaks x-axis. Default, 0.02. Only active when the `party.distortions` matrix output of the `distortion` function is to be plotted.

`labels.y.size` Size to be used for the labels of the y axis when `type = "districts"`. Default, 9.

`labels.x.size` Size to be used for the labels of the x axis when `type = "districts"`. If missing it is determined as a function of the number of parties to be plotted.

`parties` A vector with the indexes of the parties (columns) to be plotted. If missing, the six parties with the highest support in the election are chosen. Only active when an output of the `distortion` function is to be plotted.

`party.names` A vector with the names of the parties. If missing, names of the parties are inferred from the names of the columns to be plotted. Only active when an output of the `distortion` function is to be plotted.

`panels.title` A vector of length seven (for outputs of `distortion`) or of length one (for outputs of `inequality`) with the titles of the panels. If missing, they are automatically assigned taking into account the plotted component in each panel.

`axis.title` A vector of length two with the titles of the axis. If missing, they are automatically assigned taking into account the variable plotted in each axis.

...	Other arguments passed on to methods.
show.plot	A TRUE/FALSE argument indicating if the plot should be displayed as a side-effect. By default, TRUE.

Value

Invisibly returns the (ggplot) description of the plot, which is a list with components that contain the plot itself, the data, information about the scales, panels, etc.

Note

The packages ggplot2 and gridExtra need to be installed for this function to work.

Author(s)

Jose M. Pavia, <pavia@uv.es>

Alberto Penades, <alberto.penades@gmail.com>

Examples

```
votes.ex <- structure(list(UCD = c(92019L, 112927L, 117482L, 152498L, 89494L,
                                   103697L, 115390L, 223252L, 55837L, 46820L,
                                   138575L, 177843L, 163536L, 187254L, 140237L,
                                   102719L, 64603L, 102362L, 70550L, 39321L,
                                   101891L, 71650L, 89363L, 137495L, 51568L,
                                   108862L, 50270L, 35324L, 104145L, 60626L,
                                   355857L, 45231L, 45191L, 69014L, 148697L,
                                   115729L, 221996L, 87059L, 99440L, 198231L,
                                   737699L, 75036L, 38338L, 0L, 91262L, 181633L,
                                   56917L, 197100L, 84115L, 334705L, 8808L, 10723L),
                           PSOE = c(50723L, 150896L, 122361L, 111746L, 63172L,
                                    123708L, 185095L, 251336L, 33693L, 16423L,
                                    111293L, 182850L, 73554L, 39616L, 50951L,
                                    67611L, 56332L, 79715L, 28350L, 17133L, 83956L,
                                    14984L, 44388L, 64766L, 25878L, 44168L, 18210L,
                                    10757L, 76308L, 26225L, 721880L, 60747L, 27809L,
                                    59926L, 107797L, 54827L, 78598L, 20808L, 21079L,
                                    54642L, 731380L, 54720L, 34244L, 93010L, 140643L,
                                    155871L, 36186L, 213242L, 69976L, 395211L,
                                    7886L, 5186L),
                            PCE = c(11926L, 41809L, 59668L, 33879L, 10217L, 29847L,
                                    50990L, 91914L, 7398L, 2442L, 22446L, 60297L,
                                    14046L, 7467L, 10643L, 13971L, 13623L, 15562L,
                                    7927L, 5521L, 21942L, 2406L, 4765L, 12460L, 3744L,
                                    5522L, 2101L, 1196L, 15798L, 2783L, 469361L,
                                    24746L, 22680L, 41345L, 21932L, 6797L, 16777L,
                                    2949L, 2627L, 11835L, 247038L, 6319L, 3906L,
                                    12042L, 29968L, 29840L, 3846L, 50444L, 14029L,
                                    106133L, 0L, 966L),
                             AP = c(14886L, 20424L, 33616L, 24573L, 9415L, 26422L,
                                    34838L, 42860L, 7223L, 15180L, 34857L, 77932L,
```

```
                                    28472L, 15944L, 27828L, 36598L, 16005L, 31310L,
                                    10277L, 12815L, 44091L, 7123L, 28707L, 33285L,
                                    14638L, 15259L, 7407L, 3792L, 20521L, 30677L,
                                    75097L, 7953L, 10067L, 15216L, 21880L, 19516L,
                                    50256L, 36377L, 21502L, 40104L, 242077L, 0L,
                                    7927L, 27048L, 36934L, 30167L, 19925L, 35755L,
                                    14596L, 60410L, 2915L, 2074L)),
                                class = "data.frame", row.names = c(NA, -52L))

seats.ex <- structure(list(UCD = c(3L, 2L, 3L, 4L, 3L, 3L, 3L, 5L, 2L, 2L, 3L,
                                    4L, 4L, 5L, 5L, 3L, 2L, 3L, 3L, 2L, 2L, 3L,
                                    3L, 4L, 2L, 3L, 2L, 3L, 3L, 2L, 5L, 1L, 1L,
                                    2L, 4L, 4L, 6L, 4L, 4L, 6L, 11L, 3L, 2L, 0L,
                                    2L, 4L, 2L, 4L, 2L, 5L, 1L, 1L),
                           PSOE = c(2L, 4L, 3L, 3L, 2L, 4L, 4L, 5L, 1L,
                                    1L, 3L, 4L, 2L, 1L, 2L, 1L, 2L, 2L, 1L, 1L,
                                    2L, 0L, 1L, 1L, 1L, 1L, 1L, 0L, 2L, 1L, 11L,
                                    2L, 1L, 1L, 3L, 1L, 2L, 0L, 0L, 1L, 11L, 2L,
                                    1L, 3L, 3L, 4L, 1L, 4L, 2L, 7L, 0L, 0L),
                           PCE = c(0L, 1L, 1L, 0L, 0L, 0L, 1L, 2L, 0L, 0L, 0L,
                                    1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
                                    0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 7L, 0L, 0L,
                                    1L, 0L, 0L, 0L, 0L, 0L, 0L, 4L, 0L, 0L, 0L,
                                    0L, 0L, 0L, 1L, 0L, 1L, 0L, 0L),
                           AP = c(0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
                                   1L, 0L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 1L, 0L,
                                   0L, 1L, 0L, 0L, 0L, 0L, 0L, 1L, 1L, 0L, 0L,
                                   0L, 0L, 0L, 1L, 1L, 1L, 1L, 3L, 0L, 0L, 0L,
                                   1L, 0L, 1L, 0L, 0L, 1L, 0L, 0L)),
                           class = "data.frame", row.names = c(NA, -52L))
 census.ex <- c(239935L, 559590L, 451021L, 456952L, 255557L, 403142L, 569535L,
                842042L, 153109L, 112376L, 547396L, 783206L, 407899L, 365063L,
                397732L, 327008L, 211249L, 310547L, 151099L, 95061L, 309595L,
                130998L, 238885L, 364127L, 125385L, 241301L, 99408L, 73992L,
                295410L, 164706L, 3004988L, 304448L, 242148L, 331744L, 413193L,
                279372L, 731499L, 302293L, 315033L, 548711L, 2744152L, 319222L,
                153080L, 437592L, 749572L, 551021L, 167030L, 677156L, 283161L,
                1287981L, 31672L, 25368L)

example <- distortion(votes.ex, seats.ex, census.ex)
p <- plot(example, show.plot = FALSE)
p
```

`traditional_indexes` *Traditional indexes for measuring deviations from proportionality*

Description

Computes a list of indexes proposed in the literature to measure the seat-to-vote deviations from proportionality. (See the `malapportionment_index` function for the Samuels and Snyder's index).

Usage

```
traditional_indexes(votes, seats)
```

Arguments

votes	A vector of length p (where p the number of parties) with the votes gained by each party in the election. When blank or null votes are included in votes, they are treated in the same way as the other parties.
seats	A vector of length p (where p the number of parties) with components in the same order as votes, with the seats gained by each party/candidacy in the election.

Value

A data.frame of order 10x3 with the Rae, Loosemore-Handy, Gallagher, Monroe, Sainte-Lague, Entropy, Taagepera-Laasko-v, Taagepera-Laasko-s, Taagapera-Shugart, Pessini-Gini and Pavia-Gini indexes.

Note

The Entropy index (EI) only considers parties gaining seats.

Author(s)

Alberto Penades, <alberto.penades@gmail.com>

Jose M. Pavia, <pavia@uv.es>

References

Riera, P and Penades, A *Indices of Disproportionality*. Elgar Encyclopedia of Political Representation (Eds. A Freire, A Pedrazzani & H Schmit). Cheltenham: Edgar Elgar Publishing.

Taagepera, R and Shugart, MS (1991) *Seats and Votes: The Effects and Determinants of Electoral Systems*. New Haven: Yale University Press.

Examples

```
votes.ex <- c(Party1 = 6310391, Party2 = 5371866, Party3 = 1709890,
              Party4 = 1504771, Party5 = 323444)

seats.ex <- c(Party1 = 165, Party2 = 118, Party3 = 20,
              Party4 = 16, Party5 = 0)

example <- traditional_indexes(votes.ex, seats.ex)
example
```

Index